Homework 1 Solutions of CS4550, Summer 2002

2. Consider sending a large file of F bits from Host A to Host B. There are two links and one switch between A and B, and the links are uncongested (that is, no queuing delays). Host A segments the file into segments of S bits each and adds 40 bits of header to each segment, forming packets of $L = 40 + S$ bits. Each link has a transmission rate of R bps.
   (a) (5 points) Give the expression for total delay of moving the file from Host A to Host B. Neglect propagation delay and any processing delay.

<u>Answer:</u>

The transmission delay of one packet at each link is: $(S + 40)/R$ (seconds).
To figure out the total delay, just need to derive when the last packet arrives at B, relative to the time when the first packet leaves A.
The total delay is thus (total # of packets + 1) * $(S + 40)/R$
$= (F/S + 1) * (S + 40) / R$.

(b) (5 points) Find the value of S that minimizes this delay.
Taking the derivative of the expression derived in part (a) and setting it to 0, we have
$\{(F/S + 1)' * (S + 40) / R\} + \{(F/S + 1) * (S + 40)' / R\} = 0$
$\Rightarrow - F/S^2 * (S + 40) + (F/S + 1) = 0$
$\Rightarrow (S^2 – 40F) / S^2 = 0$
$\Rightarrow S = \mathrm{sqrt}(40F)$.
Taking the second derivative, we have
$((F/S + 1) * (S + 40) / R)'' = ((S^2 – 40F) / S^2)' = 80F / S^3$.
Plug in the sqrt (40F) for S, we obtain a positive value. This confirms that S = sqrt (40F) minimizes the total delay.

3. Suppose client A initiates an FTP session with server S. At about the same time, client B also initiates an FTP session with the same server. Provide possible source and destination ports for
   (a) (3 points) the segments sent from A to S.

<u>Answer:</u>

FTP Control port: 21, Data port: 20

(source-port, destination-port) = (X, 21) where X could be any value between 1024 and 65535.

(b) (2 points) the segments sent from B to S.

<u>Answer:</u>

(source-port, destination-port) = (Y, 21) where Y could be any value between 1024 and 65535. An additional constraint on Y is discussed in part (f).

(c) (3 points) the segments sent from S to A.

<u>Answer:</u>

(source-port, destination-port) = (20, X).

(d) (2 points) the segments sent from S to B.

<u>Answer:</u>

(source-port, destination-port) = (20, Y).

(e) (3 points) If A and B are at different hosts, is it possible that the source port numbers in the segments from A to S are the same as those from B to S?

Answer:

Yes. The server will use the client host addresses to distinguish the two clients.

(f) (2 points) How about if they are at the same host?

Answer:

No. The OS must choose two different ports for the two clients.

4. Consider the TCP procedure for estimating RTT. Suppose that x = 0.1. Let SampleRTT_1 be the most recent sample RTT. Let SampleRTT_2 be the next most recent sample RTT, etc.
   (a) (5 points) For a given TCP connection, suppose 4 acknowledgements have been returned with corresponding sample RTTs SampleRTT_4, SampleRTT_3, SampleRTT_2, and SampleRTT_1. Express EstimatedRTT in terms of the four sample RTTs.

Answer:

Denote SampleRTT_i by y_i, $i = 1,2,\ldots$ Assume that *EstimatedRTT* is initially 0.

Using the first sample (y_4),

$EstimatedRTT = (1\text{-}x)(initial\ EstimatdRTT) + (x)\ y_4 = 0.1y_4$

Using the second sample (y_3),

$EstimatedRTT = (1\text{-}x)\ EstimatdRTT + (x)\ y_3 = 0.9(0.1\ y_4) + 0.1\ y_3$

Using the third sample (y_2),

$EstimatedRTT = (1\text{-}x)\ EstimatdRTT + (x)\ y_2 = 0.9(0.9(0.1\ y_4) + 0.1\ y_3) + 0.1\ y_2$

Using the last sample (y_1),

$EstimatedRTT = 0.9(0.9(0.9(0.1\ y_4) + 0.1\ y_3) + 0.1\ y_2) + 0.1\ y_1$

$= 0.0729\ y_4 + 0.081\ y_3 + 0.09\ y_2 + 0.1\ y$

(b) (5 points) Generalize your formula for n sample RTTs.

Answer:

Generalized the previous formula to n samples. Let y_1 be the most recent sample, etc.

$$EstimatedRTT = x\ ((1\text{-}x)^{n-1}y_n + (1\text{-}x)^{n-2}y_{n-1} + \ \ldots + (1\text{-}x)\ y_2 + y_1) = x\sum_{i=1}^{n}(1-x)^{i-1}y_i\ .$$

In this case, it is equal to

$0.1\ ((0.9)^{n-1}y_n + (0.9)^{n-2}y_{n-1} + \ \ldots + (0.9)\ y_2 + y_1).$

(c) (5 points) Let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.

Answer:

From the formula derived in part (b), it is clear that the effect of an old SampleRTT decays exponentially. In other words, the *EstimatedRTT* moves away from the old sample at an exponential pace.

5. You are hired to design a reliable byte-stream protocol that uses a sliding window (like TCP). This protocol will run over a 100-Mbps network. The RTT of the network is 100 ms, and maximum segment lifetime is 60 seconds.
   (a) (5 points) How many bits would you include in the RcvWindow and SequenceNumber fields of your protocol header?

Answer:

From the assumptions, the maximum number bits to fill the pipe (up to delay bandwidth product) is

(R * RTT) = 100Mbps * 0.1 seconds = 10 Mb.

Therefore, the maximum receiver window size is 10 megabits = 1250,000 bytes. If the RcvWindow is measured by bytes, then the number of bits needed for the RcvWindow should be at least *ceiling*(log_2*(1250000))* = 21 bits.

Delay bandwidth product – sender can send this much before receiving feed back from receiver. If the receiver decides to stop sender from transmitting, it still needs to buffer this amount of bytes.

The sequence number field should be big enough so that it will not wrap around too early causing two different segments to have the same sequence number. Therefore, the number of bits for SequenceNumber should be at least:

ceiling(log_2*(R * packet-lifetime)) = ceiling*(log_2*(750,000,000)) = 30.*

(b) (5 points) How would you determine the numbers given above, which values might be less certain?

Answer:

The packet lifetime is a fixed network design parameter. Therefore, the number of bits for SequenceNumber is very certain. The RTTs in the network may change dramatically. Therefore, more bits should be used for RcvWindow to allow some margin of errors.

6. (10 points) Suppose, in TCP's RTT estimation mechanism, that EstimatedRTT is 4.0 ms at some point and subsequent sampled RTTs all are 1.0 ms. How long does it take before the Timeout value falls below 4.0 ms? Assume a plausible initial value of Deviation; how sensitive is your answer to this choice? Use x = (1/8).

Answer:

Take deviation to be 2.0 ms. It takes 21 samples before Timeout falls below 4.0 ms. The answer is not very sensitive to the initial deviation value. It takes 19 samples if the initial deviation is 0.5 ms and 22 samples for an initial deviation of 3.0 ms.

7. Assume that TCP implements an extension that allows window sizes much larger than the current maximum of 64Kbytes. Suppose you are using this extended TCP over a 1-Gbps link with a latency of 100 ms to transfer a 10-MB file, and the receiver has buffer space of 1MB. If TCP sends 1-KB segments (excluding TCP header), there is no congestion and no lost packet, and the receiving application reads segments as fast as they come in:

(a) (5 points) How many RTTs does it take until slow start opens the send window to 1MB?

Answer:

Assume that the initial congestion window threshold is set to 1 MB or greater. Then it takes *ceiling*(log_2*(1024)) = 10* RTTs to open the congestion window to 1MB.

(*bonus) If the initial congestion window threshold is set to be x KB, which is less than 1 MB, then it takes *ceiling*(log_2*(x)) + (1024 - x)* RTTs.

(b) (5 points) How many RTTs does it take to send the file?

Answer:

Since the receive buffer size is 1MB, the maximum effective window size is 1MB regardless of congestion window. From transmission rounds 1 to 10, a total of (1+2+...+2^9) = 1MB-1KB of data is sent. Therefore, it takes another 10 RTTs to send the remaining 9MB + 1KB of the 10-MB file. So the answer is 20.

(c) (5 points) If the time to send the file is given by the number of required RTTs multiplied by the link latency, what's the effective throughput, (Total Bytes Sent / Send Time), for the transfer? What percentage of the link bandwidth is utilized?

Answer:

Total send time = 20 * 100 ms = 2.0 seconds.

Effective throughput = Total bytes sent / Total send time = (10 * 2^20) / 2

= 5242880 bytes/s = 5242880 * 8 (bps) = 41943040 (bps)

Therefore, link utilization = Effective throughput / Total bandwidth

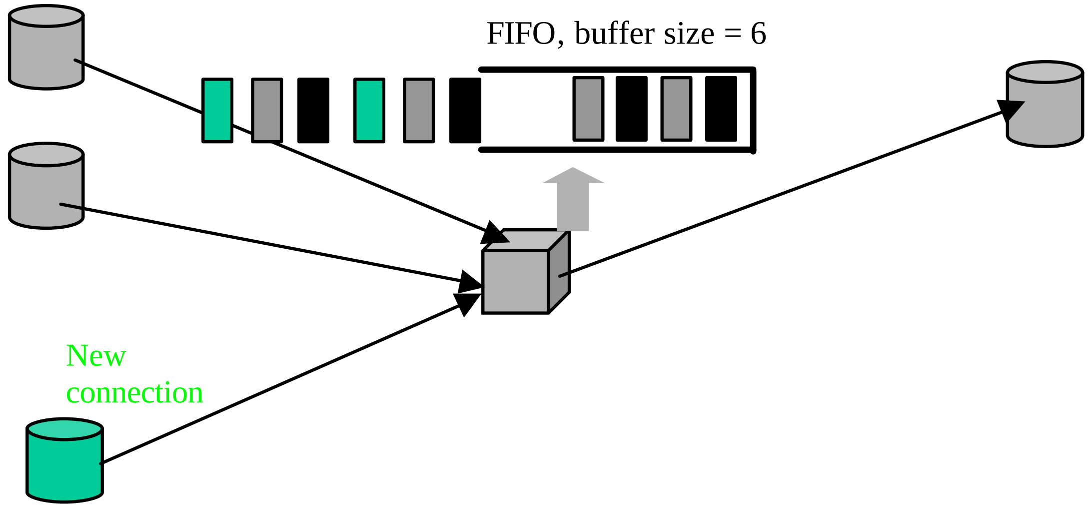
= 41,943,040 / 1,000,000,000 = 0.042 = 4.2%.

8. Suppose two TCP connections share one of router R's links. The queue size for the shared link is six segments; each connection has a stable congestion window of three segments. No congestion control is used by these connections. A third TCP connection now is attempted, also using the same link. The third connection does not use congestion control either.

(a) (5 points) Describe a scenario in which, for at least a while, the third connection gets none of the available bandwidth, and the first connections proceed with 50% each.

Answer:

The following hypothetical scenario could happen. Initially the first two connections fill the buffer with 3 packets each. Then the two connections start to send constant rate traffic with the rate equal to half of the link capacity. For some period of time, each packet of the third connection may arrive just behind a new burst of packets from the first two connections and get dropped because the buffer is full. This scenario is more likely when the RTT time of third connection is much larger than the other two connections.



(b) (5 points) Does it matter if the third connection uses slow start?

Answer:

No. The problem is caused by the other two connections utilizing the entire bandwidth. Slowing down the third connection will not eliminate the problem.